TonUP Smart Contract Audit Report

Fri May 24 2024

contact@bitslab.xyz https://twitter.com/tonbit_

TonBit

TonUP Smart Contract Audit Report

1 Executive Summary

1.1 Project Information

Description	A launchpad and staking project.
Type	Launchpad
Auditors	TonBit
Timeline	Thu May 23 2024 - Fri May 24 2024
Languages	Tact
Platform	Ton
Methods	Architecture Review, Unit Testing, Manual Review
Source Code	https://github.com/nx-fi/tonup
Commits	c769ecc750a016491fde081e845531e360c25ded d923299211754e26959d1729592b67168813629a

1.2 Files in Scope

The following are the SHA1 hashes of the original reviewed files.

ID	File	SHA-1 Hash
TLFP	contracts/tonup_launcher_fixed_price.tact	09e301cf4b805fa4da1e03d6a8d88c2e647c3386
TWH	contracts/tonup_whitelist.tact	bfda38cd5334bec439bfdd98b46a53691403218d
TST	contracts/tonup_staking.tact	85c20e2037f73d6ea36ccc559e96239cec72a177
TTL	contracts/tonup_token_locker.tact	fc64e3e2c314f602e12b27b344ce4759fc3c174a

1.3 Issue Statistic

Item	Count	Fixed	Acknowledged
Total	6	6	0
Informational	1	1	0
Minor	3	3	0
Medium	2	2	0
Major	0	0	0
Critical	0	0	0

1.4 TonBit Audit Breakdown

TonBit aims to assess repositories for security-related issues, code quality, and compliance with specifications and best practices. Possible issues our team looked for included (but are not limited to):

- Transaction-ordering dependence
- Timestamp dependence
- Integer overflow/underflow by bit operations
- Number of rounding errors
- Denial of service / logical oversights
- Access control
- Centralization of power
- Business logic contradicting the specification
- Code clones, functionality duplication
- Gas usage
- Arbitrary token minting
- Unchecked CALL Return Values

1.5 Methodology

The security team adopted the **"Testing and Automated Analysis"**, **"Code Review"** strategy to perform a complete security test on the code in a way that is closest to the real attack. The main entrance and scope of security testing are stated in the conventions in the "Audit Objective", which can expand to contexts beyond the scope according to the actual testing needs. The main types of this security audit include:

(1) Testing and Automated Analysis

Items to check: state consistency / failure rollback / unit testing / value overflows / parameter verification / unhandled errors / boundary checking / coding specifications.

(2) Code Review

The code scope is illustrated in section 1.2.

(3) Audit Process

- Carry out relevant security tests on the testnet or the mainnet;
- If there are any questions during the audit process, communicate with the code owner in time. The code owners should actively cooperate (this might include providing the latest stable source code, relevant deployment scripts or methods, transaction signature scripts, exchange docking schemes, etc.);
- The necessary information during the audit process will be well documented for both the audit team and the code owner in a timely manner.

2 Summary

This report has been commissioned by TonUP to identify any potential issues and vulnerabilities in the source code of the TonUP smart contract, as well as any contract dependencies that were not part of an officially recognized library. In this audit, we have utilized various techniques, including manual code review and static analysis, to identify potential vulnerabilities and security issues.

During the audit, we identified 6 issues of varying severity, listed below.

ID	Title	Severity	Status
TLF-1	Missing `distribution_complete` Check	Medium	Fixed
TLF-2	There May Be Unexpected Abort	Medium	Fixed
TLF-3	Missing Time Check	Minor	Fixed
TLF-4	Incorrect Event Emit	Minor	Fixed
TLF-5	Unintuitive Judgment Conditions	Minor	Fixed
TLF-6	Code Optimization	Informational	Fixed

3 Participant Process

Here are the relevant actors with their respective abilities within the TonUP Smart Contract :
Admin

- The `Admin` can send message `SetStakingConfig` to set `StakingConfig` .
- The `Admin` can send message `SetMetadata` to set `METADATA_NAME` .
- The `Admin` can send message `enable claim` to make `self.config.reward_is_claimable` true and claim function available.
- The `Admin` can send message `disable claim` false and claim function not available.
- The `Admin` can send message `enable unstake` to make `self.config.unstake_during_reward_period` true and unstake function available.
- The `Admin` can send message `AddReward` to add new reward epoch configration.
- The `Admin` can send message `withdraw dust` to withdraw assets stucked in the contract.
- The `Admin` can send message `RescueTokens` to send assets that is not `staking_token_wallet_address` from the contract.
- The `Admin` can send message `SetLauncherConfig` to set launch config.
- The `Admin` can send message `SetMetadata` to set metadata.
- The `Admin` can send message `SetTokenWalletAddress` to set `self.token_wallet_address` .
- The `Admin` can send message `SetUpWalletAddress` to set `self.up_wallet_address` .
- The `Admin` can send message `AddDistributionPool` to add new distribution pool.
- The `Admin` can send message `DeleteDistributionPool` to delete the distribution pool.
- The `Admin` can send message `ModifyDistributionPool` to modify the distribution pool.
- TThe `Admin` can send message `ManualTransferNotification` to increase `self.tokens_awaiting_launch` .
- The `Admin` can send message `SetWhitelistContract` to set the white list address.

- The `Admin` can send message `validate` to make `self.config_validated` true.
- The `Admin` can send message `finalize` to refund any unsold tokens to the unsold token refund address.
- The `Admin` can send message `distribute` to distribute tons according to pool settings.
- The `Admin` can send message `withdraw dust` to withdraw assets stuck in contract.
- The `Admin` can send message `RescueJetton` to send assets that is not `staking_token_wallet_address` from contract.
- The `Admin` can send message `SetWhitelistConfig` to set the white list config.
- The `Admin` can send message `AddDataWriter` to add data writer.
- The `Admin` can send message `RemoveDataWriter` to remove data writer.
- The `Admin` can send message `BatchWhitelistUser` to batch add/remove users to/from the whitelist.
- The `Admin` can send message `WhitelistUserParticipateCommunication` to return full amount to `msg.owner` or `self.launcher`.

User

- The `User` can send assets to specified wallet and wallet would send message `JettonTransferNotification` to stake.
- The `User` can send message `Ustake` to unstake assets.
- The `User` can send message `claim` to claim rewards.
- The `User` can send message `Participate` to participate in the launch.
- The `User` can send message `claim` to claim rewards.
- The `User` can send message `refund` to refund their tons in case launch failed.
- The `User` can send message `withdraw` to initiate a withdraw to the recipient.

4 Findings

TLF-1 Missing `distribution_complete` Check

Severity: Medium

Status: Fixed

Code Location:

contracts/tonup_launcher_fixed_price.tact#465

Descriptions:

The lack of checking of the `distribution_complete` variable in the `launch_successful` condition resulted in the `distribute` function being able to be called repeatedly.

Suggestion:

It is recommended to add a check `self.distribution_complete != false` in the `self.launch_successful` condition.

Resolution:

This issue has been fixed. The client has adopted our suggestions.

TLF-2 There May Be Unexpected Abort

Severity: Medium

Status: Fixed

Code Location:

contracts/tonup_launcher_fixed_price.tact#502

Descriptions:

In the `withdraw dust` function, the calculation of the `dust_to_claim` variable may result in an unexpected abort due to insufficient subtraction of `myBalance` because `total_ton_contributed` has been distributed and the amount of `myBalance` will be reduced.

Suggestion:

It is recommended to add a check for the `dust_to_claim` .

Resolution:

This issue has been fixed. The client has adopted our suggestions.

TLF-3 Missing Time Check

Severity: Minor

Status: Fixed

Code Location:

contracts/tonup_launcher_fixed_price.tact#331

Descriptions:

`start_time` and `end_time` are not checked in the `SetLauncherConfig` function.

Suggestion:

It is recommended to add a check as:

```
require(phase_config.start_time >= config.end_time, "Invalid claim start time");
```

Resolution:

This issue has been fixed. The client has adopted our suggestions.

TLF-4 Incorrect Event Emit

Severity: Minor

Status: Fixed

Code Location:

contracts/tonup_launcher_fixed_price.tact#361,370

Descriptions:

The event emit in the `SetTokenWalletAddress` function and the `SetUpWalletAddress` function does not match the function's functionality.

Suggestion:

It is recommended to use coorrect event emit.

Resolution:

This issue has been fixed. The client has adopted our suggestions.

TLF-5 Unintuitive Judgment Conditions

Severity: Minor

Status: Fixed

Code Location:

contracts/tonup_launcher_fixed_price.tact#577

Descriptions:

In the conditional judgment of the claim function, here is:

```
require(self.total_ton_contributed >= self.config.min_total_ton, "Min TON limit not met");
```

But in the function finalize:

```
if (self.total_ton_contributed < self.config.min_total_ton) {
            self.launch_successful = false;
            tokens_to_refund = self.tokens_awaiting_launch;
        } else {
            self.launch_successful = true;
            tokens_to_refund = self.tokens_awaiting_launch - self.total_ton_contributed *
self.config.tokens_per_ton / ton("1");    // Very unlikely to overflow (need 1e59 tokens)
        }
```

It has determined the size of these two values.

Suggestion:

It is recommended to use `self.launch_successful` to determine whether it is successful to finalize.

Resolution:

This issue has been fixed. The client has adopted our suggestions.

TLF-6 Code Optimization

Severity: Informational

Status: Fixed

Code Location:

contracts/tonup_launcher_fixed_price.tact#590

Descriptions:

When there are no new unlocks to be extracted(`jetton_claim_amount - already_claimed = 0`), the transaction should be abort to avoid initiating multiple transactions.

Suggestion:

It is recommended to add an assert when `jetton_claim_amount - already_claimed = 0` .

Resolution:

This issue has been fixed. The client has adopted our suggestions.

Appendix 1

Issue Level

- **Informational** issues are often recommendations to improve the style of the code or to optimize code that does not affect the overall functionality.
- **Minor** issues are general suggestions relevant to best practices and readability. They don't post any direct risk. Developers are encouraged to fix them.
- **Medium** issues are non-exploitable problems and not security vulnerabilities. They should be fixed unless there is a specific reason not to.
- **Major** issues are security vulnerabilities. They put a portion of users' sensitive information at risk, and often are not directly exploitable. All major issues should be fixed.
- **Critical** issues are directly exploitable security vulnerabilities. They put users' sensitive information at risk. All critical issues should be fixed.

Issue Status

- **Fixed:** The issue has been resolved.
- **Partially Fixed:** The issue has been partially resolved.
- **Acknowledged:** The issue has been acknowledged by the code owner, and the code owner confirms it's as designed, and decides to keep it.

Appendix 2

Disclaimer

This report is based on the scope of materials and documents provided, with a limited review at the time provided. Results may not be complete and do not include all vulnerabilities. The review and this report are provided on an as-is, where-is, and as-available basis. You agree that your access and/or use, including but not limited to any associated services, products, protocols, platforms, content, and materials, will be at your own risk. A report does not imply an endorsement of any particular project or team, nor does it guarantee its security. These reports should not be relied upon in any way by any third party, including for the purpose of making any decision to buy or sell products, services, or any other assets. TO THE FULLEST EXTENT PERMITTED BY LAW, WE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, IN CONNECTION WITH THIS REPORT, ITS CONTENT, RELATED SERVICES AND PRODUCTS, AND YOUR USE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NOT INFRINGEMENT.